임베디드 시스템 교육을 위한 가상 실습 키트

Virtual Experimental Kit for Embedded System Education

조상영
한국외국어대학교 컴퓨터공학과

Sang-Young Cho(sycho@hufs.ac.kr)

요약

임베디드 시스템 과목을 위한 실습 과제는 주로 임베디드 보드와 소프트웨어 개발 도구를 사용한 하드웨어 실습 키트로 진행된다. 하드웨어 실습 키트는 높은 초기 설치비용, 유지보수의 어려움, 산업계 발전에 비적응적 대처, 교육적 성과의 한계와 같은 단점을 가지고 있다. 본 논문은 임베디드 시스템 하드웨어 실습 키트의 단점을 극복할 수 있는 시뮬레이션 기반의 가상 실습 환경의 사용을 제안하고 가상 실습 키트의 설계 및 구축에 대하여 기술한다. 구축된 가상 실습 키트는 ARM 사의 ARMulator 환경에 기반을 두어 마이크로프로세서 시스템의 주요 하드웨어 IP들을 추가하고 주변장치들을 위한 사용자 인터페이스 모듈을 개발하여 구축되었다. 검증용 예제 프로그램을 이용하여 동작의 정확성을 확인하였으며 실시간 운영체제 실습도 가능하도록 MicroC/OS-II를 이식하였다.

■ 중심어 : | 임베디드 시스템 교육 | 실습 키트 | 시뮬레이션 | 가상 환경 |

Abstract

Laboratory works for embedded system courses are usually performed with hardware based experimental kits that equipped with an embedded board and software development tools. Hardware-based kits have demerits such as high initial setup cost, burdensome maintenance, inadaptability to industry evolution, and restricted educational outcomes. This paper proposes using virtual experimental environments to overcome the demerits of hardware-based kits and describes the design and implementation of a simulation-based virtual experimental kit. With ARM's ARMulator, we developed the kit by adding hardware IPs and user interface modules for peripherals. The developed kit is verified with an experimental program that uses all the augmented software modules. We also ported MicroC/OS-II on the virtual experimental kit for real-time OS experiments.

■ keyword : | Embedded System Education | Experimental Kit | Simulation | Virtual Environment |

I. 서 론

임베디드 시스템의 교육은 하드웨어, 소프트웨어, 개발 환경 등 다양한 컴퓨터 관련 영역을 포함하고 있다. 특히, 제약된 자원 환경에서의 시스템 성능, 어셈블리언어 또는 고급언어를 이용한 하드웨어 제어, 다양한 주변 장치에 대한 이해, 하드웨어/소프트웨어 개발 환경의 숙달, 실시간 운영체제에 대한 학습은 임베디드 시

* 본 연구는 2009년 한국외국어대학교 교내연구비 지원에 의하여 수행되었습니다
접수번호 : #091124-006
접수일자 : 2009년 11월 24일
심사완료일 : 2010년 01월 18일
교신저자 : 조상영, e-mail : sycho@hufs.ac.kr

스템 개발자를 양성하기 위한 교육의 필수적인 요소들이다[1][2]. 이러한 주제들은 정확히 이해하기 위해서는 강의 및 교재에서 제공되는 이론 및 내용에 대한 이해뿐만 아니라 시스템의 구성 및 동작에 대한 심도 있는 이해를 위한 실제적인 실습이 필요하다. 보통 임베디드 시스템 관련 과목의 실습은 임베디드 시스템 보드, 소프트웨어 개발 환경, 실습 예제로 구성되는 실습 키트를 이용하며 때로는 산업 현장에서 사용하는 고급 디버깅 장비가 사용된다[3][4].

하드웨어 키트를 사용하는 임베디드 시스템 실습은 학습자가 직접적인 체험을 통해 실제 개발 환경을 이해하고 시스템의 동작을 정확히 파악하는데 도움을 주는 반면에 유지보수 및 교육적 성과 측면에서 다음과 같은 단점을 가지고 있다[4-6]. 첫째, 숙련되지 않은 학생들이 실습 과제를 수행하는 도중에 부주의 또는 잘못된 조작으로 하드웨어 키트의 고장을 초래하는 경우가 많다. 둘째, 하드웨어 실습 키트는 고급 디버깅 장비와 최신의 소프트웨어 개발 환경을 포함할 경우에 매우 고가이며 실습을 위한 운용 컴퓨터와 각종 배선을 필요로 한다. 셋째, 임베디드 시스템 제품은 시장에서의 경쟁이 치열하며 기술의 발전 및 적용이 급격하게 이루어지고 있다. 산업계의 기술 수요에 적합한 인력을 양성하기 위해서는 하드웨어 실습 환경의 지속적인 갱신이 필요하다. 넷째, 하드웨어 실습 보드는 주어진 입력에 대하여 블랙박스 형태의 내부 동작을 거쳐 출력으로만 결과를 확인할 수 있다. 따라서 모듈 내부의 자세한 동작이나 성능에 관련된 자세한 데이터를 제공하지 못한다.

시뮬레이터를 이용한 실습 환경은 하드웨어 키트를 사용하는 환경의 단점들을 극복할 수 있기 때문에 디지털 논리회로와 컴퓨터구조와 같은 컴퓨터 하드웨어에 관련 과목의 교육용 하드웨어 시뮬레이터가 많이 개발되었다[7][8]. [7]에서는 53개의 교육용 시뮬레이터를 조사하여 특화되어 있는 주제별로 분류하였고 [8]에서는 28개의 시뮬레이터를 조사하여 IEEE의 컴퓨터구조 과목의 학습내용의 커버리지를 구하였다. 이들 조사에 의하면 대부분 교육용 시뮬레이터들은 하드웨어 특정 주제에 대해 특화되어 하드웨어 관련 과목 전반에 걸쳐 사용하거나 임베디드 시스템과 같이 다양한 하드웨어 및 소프트웨어 관련된 주제를 다루기에 어려움이 있다. 특정 주제이외에 성능 분석[5], 사용자 인터페이스[6], 웹 접근성[9] 등의 기능성에 중점을 둔 시뮬레이터들도 있으나 기존의 대부분의 시뮬레이터들은 임베디드 시스템 실습에 필수적인 주변장치 제어에 대한 실습을 거의 다루고 있지 않다.

기존 시뮬레이터의 단점을 해결하기 위하여 컴퓨터 구조, 운영체제, 컴파일러 과목에서 사용할 수 있는 시뮬레이터[10]가 시도되었으나 개념 교육을 위한 단순한 아키텍처를 구현하였다. 다양한 분야를 위한 임베디드 시스템의 통합된 실습을 위한 환경의 개발은 아직 하드웨어 단계에 머물고 있다[3][4].

본 논문에서는 하드웨어 실습 키트의 단점을 극복하면서도 다양한 임베디드 시스템 교육의 주제들을 모두 다룰 수 있는 통합된 실습 환경을 위하여 가상 개발 환경(VDE: Virtual Development Environment)에 기반을 둔 가상 실습 키트의 사용을 제안한다. 임베디드 시스템 과목을 위한 하드웨어 시뮬레이터는 기존의 시뮬레이터에 비하여 더 일반적이고 강력하면서도 실제 하드웨어 동작을 더 정확히 관찰할 수 있도록 구현되어야 한다. 임베디드 시스템 개발에 사용되는 VDE는 사이클 단계의 정밀한 시뮬레이션을 통하여 실제 하드웨어가 있는 것과 같은 유사한 환경 하에서 개발하고자 하는 임베디드 시스템의 다양한 하드웨어 프로토타입의 기능 및 성능을 탐색할 수 있으며 실제 하드웨어 없이 소프트웨어를 개발할 수 있거나 하드웨어와 소프트웨어의 동시 설계를 가능하게 한다[11-13]. 기존의 VDE는 임베디드 시스템 제품 개발을 위한 전문적인 상용 도구로서 매우 고가이기 때문에 대학의 교육용으로 사용되지 않았다. 본 논문에서는 대학의 교육용으로 사용할 수 있는 VDE에 기반을 둔 가상 실습 키트를 설계하고 구현하였으며 이를 검증하였다. 특히 전체 시스템의 정확한 성능 측정을 위하여 주변장치가 사이클 단위의 시뮬레이션이 가능하도록 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 가상 개발 환경과 ARMulator 환경에 대해 기술한다. 3장에서는 가상 실습 키트의 설계와 구현, 그리고 검증에 대해 기술한다. 4장에서는 구축된 가상 실습 키

트의 활용에 대해 논의하고 마지막 5장에서 제안된 시스템의 고찰 및 결론에 대해 기술한다.

II. 관련 연구

1. 가상 개발 환경 (VDE)

가상 개발 환경은 실제 임베디드 보드의 정밀한 시뮬레이션을 위한 프로세서, 메모리, 주변장치 모델들과 시뮬레이션 엔진을 제공하며 실제 하드웨어와 유사한 사용자 인터페이스를 제공한다. 또한 소프트웨어 개발 및 수행을 위한 개발 도구와 연동될 수 있으며 이를 통해 임베디드 시스템 개발의 효율성을 높일 수 있다[11-13]. 비록 가상 개발 환경이 실제 임베디드 보드는 아니지만 환경에서 제공되는 하드웨어 IP(Intellectual Property)와 소프트웨어 IP를 이용하여 대부분의 응용 소프트웨어들이 실제 환경에서와 같이 개발되고 검증될 수 있다.

Virtual Platform[11]은 Synopsys 사의 상용 VDE로 임베디드 시스템에 많이 사용되는 ARM, X-Scale, MIPS 프로세서를 지원하며 대상 하드웨어 시스템의 소프트웨어 개발을 목표로 한다. SoC Designer[12]는 Carbon 사에서 제공하는 SystemC 기반의 SoC 개발 환경이다. 이 환경은 모델링, 시뮬레이션, 디버깅을 빠르고 쉽게 할 수 있는 도구들을 지원한다. 이 도구들을 이용하여 시스템 또는 하드웨어 개발자들은 원하는 대부분의 하드웨어를 빠르게 구성할 수 있으며 소프트웨어 개발자들은 실제 하드웨어가 만들어지기 전에 소프트웨어를 개발할 수 있다. Mentor Graphics사의 Visual Elite[13]는 'Fast ISS'라는 프로세서 모델과 TLM (Transaction Level Modelling) 기반의 버스, 메모리, 주변 장치 모델을 가지고 있어서 구성에 따라 다양한 추상 단계의 시스템을 구축할 수 있으며 소프트웨어 또는 펌웨어를 작성하고 디버깅할 수 있다. 상용 가상 개발 환경은 임베디드 시스템 제품 전문 개발자를 대상으로 하며 하드웨어 시뮬레이션 도구와 소프트웨어 개발 도구가 밀접하게 결합되어 있거나 다른 부수적인 개발 도구와 연관되어 출시되고 있다.

2. ARMulator 개발 환경

ARM 프로세서는 32-비트 RISC 프로세서로서 고성능 임베디드 시스템에서 가장 많이 사용되는 프로세서이다[4]. 특히 이동형 휴대 단말 장치의 대부분의 제어 프로세서로 사용되며 이외에도 다양한 임베디드 분야에 적용되고 있다. 현재 많은 학교에서 ARM 프로세서 기반의 실습 키트를 임베디드 시스템 관련 과목에서 사용하고 있다. 우리는 ARM사의 ARMulator를 기초 환경으로 하여 ARM920T 코어 모델을 이용하는 가상 실습 키트를 구축하였다. ARMulator는 ARM 사의 소프트웨어 개발 환경[14]인 ADS1.2 (ARM Developer Suite)와 RVDS4.0 (RealView Developer Suite)에 포함되어 있으며 ARM 프로세서를 사용하는 시스템의 대부분의 소프트웨어 개발자들은 이들 환경을 보유하고 있으며 ADS1.2의 경우는 많은 대학에서 사용하고 있다.

ARMulator는 ARM 사의 사이클-기반 ARM 프로세서 코어 명령어 집합 시뮬레이터(ISS: Instruction Set Simulator)를 근간으로 구성되어진 가상 보드 모델 및 시뮬레이션 환경이다. ARMulator 환경은 간단한 메모리 모델과 표준 하드웨어 IP 모델들을 가지고 있어서 실제 하드웨어를 제작하기 전에 소프트웨어를 개발하거나 프로파일링 기능을 이용하여 하드웨어 및 소프트웨어의 성능을 평가할 수 있다. ARMulator는 시뮬레이션 커널과 확장 모듈로 구성되어 있다. 시뮬레이션 커널은 프로세서 코어의 시뮬레이션과 RDI (Remote Debug Interface) 디버거 또는 확장 모듈의 외부 인터페이스를 담당한다. 확장 모듈은 타이머, 워치독 타이머, 타임 틱, 인터럽트 콘트롤러 등의 기본 주변 장치로 구성된다. 이러한 기본 모듈을 이용하여 기본적인 소프트웨어의 개발이 가능하지만 다양한 주변장치가 동작하는 실제 하드웨어 환경과는 차이가 많다. ARMulator는 메모리 모델의 확장 인터페이스를 이용하여 새로운 하드웨어를 모델링하여 추가할 수 있는 확장 기능을 제공하고 있다. [그림 1]은 ARMulator의 기본 가상 환경을 보여준다.

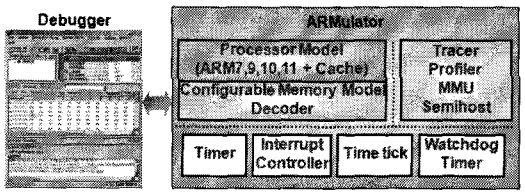


그림 1. 기본 ARMulator 환경

III. 가상 실습 키트의 설계 및 구현

1. 가상 실습 키트의 개발 방향

임베디드 시스템 교육은 하드웨어와 소프트웨어의 다양한 분야를 포함하고 있다[3]. 하드웨어로는 프로세서, 메모리, 버스, 주변 장치, 네트워크가 있으며 소프트웨어로는 펌웨어, 어셈블리언어, 고급언어, 운영체제, 응용프로그램이 있다. 하드웨어 개발을 위한 칩 또는 보드 단계의 설계 도구가 있으며 소프트웨어 개발을 위한 어셈블러, 컴파일러, 링커 및 통합 개발 환경이 있다. 때로는 하드웨어와 소프트웨어의 동시 개발을 위한 도구들이 사용되기도 한다. 본 논문에서는 개발한 가상 실습 키트는 하드웨어 개발 환경과 도구를 제외한 하드웨어, 소프트웨어, 개발환경 전 분야의 실습을 수행할 수 있도록 하였다. 그러나 다양한 하드웨어 IP들을 모두 구현할 수 없기 때문에 마이크로프로세서 시스템에서 가장 중요한 하드웨어 IP들을 선정하였으며 네트워크 관련 내용은 가상 환경의 한계로 제외하였다.

기본적으로 ARMulator 환경은 소프트웨어 개발 환경과 프로세서, 메모리 모델을 기본으로 간단한 하드웨어 IP들을 제공하고 있어서 하드웨어 제어 기초 실습과 어셈블리언어 또는 고급언어를 사용한 소프트웨어 실습이 가능하다. 그러나 제약된 환경으로 임베디드 시스템 실습 키트로 사용되기에 부족하며 다음과 같은 부분들을 고려하여 설계하였다.

1) 임베디드 시스템 실제적인 교육을 위하여 실제 산업계와 하드웨어 실습 키트에서 사용하는 프로세서를 참고하여 하드웨어 IP들을 추가한다.
2) ARMulator는 버스를 통한 하드웨어 확장 개념을 제공하지 않기 때문에 칩 내의 버스 규격인 AHB 버스와 APB 버스를 구현한다. 또한 버스에 연결되는 하드웨어 IP들의 정확한 성능 분석 기능을 위하여 사이클 단계에서 프로세서와 연동할 수 있도록 한다.
3) 교육적 효과를 높이기 위해 실제 하드웨어의 동작을 그래픽 형태로 보여 줄 수 있는 사용자 인터페이스 장치를 구현한다.
4) 실시간 운영체제 교육을 위하여 가상 실습 키트에 실시간 운영체제를 이식한다.

2. ARMulator 환경의 확장

ARMulator는 C 언어를 사용하여 DLL 파일 형태로 임의의 하드웨어를 모델링하여 추가할 수 있다. 본 논문에서는 ARM 사에서 개발된 칩 내의 버스 규격인 AHB 버스와 APB 버스, 그리고 다양한 하드웨어 IP 모듈들을 모델링하여 기존 ARMulator 환경에 추가하여 임베디드 시스템 과목을 위한 가상 실습 키트를 구축하였다.

실제 환경에 맞는 가상 실습 키트를 구현하기 위하여 임베디드 보드를 위한 SoC로 삼성의 S3C2440[15] 칩을 선택하였다. S3C2440은 모바일 장치 와 일반 용도의 응용 프로세서 (AP: Application Processor)이며 S3C2440의 하드웨어 IP 중에서 학생들의 실습에 가장 많이 사용되는 메모리 콘트롤러, LCD 콘트롤러, 인터럽트 콘트롤러, DMAC, UART, 타이머, Watchdog 타이머, GPIO 포트를 구현하였다. [그림 2]는 가상 실습 키트를 구축하기 위하여 모델링한 SoC 구성을 보여준다.

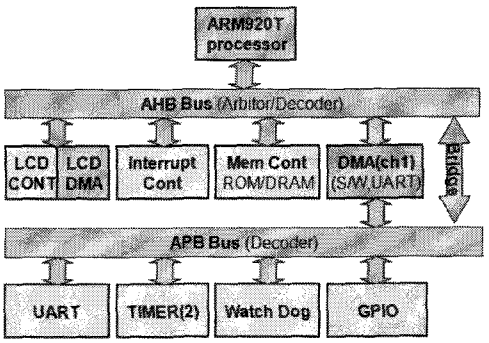


그림 2. 대상 SoC 시스템

이들 하드웨어 모델들은 구현된 AHB 버스와 APB를 이용하여 ARMulator에서 제공되는 ARM920T ISS 모델과 연결된다. ARM920T 모델은 MMU, 각각 16KB 크기의 명령어/데이터 캐시, 보조프로세서 15, AMBA 인터페이스를 포함하고 있다.

AHB 버스에 연결된 하드웨어 IP들의 모든 동작은 AHB 버스의 동작 클록인 HCLK에 동기 되도록 설계하였다. HCLK 클록은 다양한 비율로 CPU 클록에 동기 될 수 있다. 원래 ARMulator 환경에서는 하드웨어 모듈을 시뮬레이션 하기 위하여 특정 시간 경과 후에 Call-back 함수가 호출되어 모델이 동작을 수행하는 형태를 취하고 있기 때문에 하드웨어 IP들 간의 연동 동작의 처리가 쉽지 않다. 우리의 가상 실습 키트에서는 각 하드웨어 IP들이 HCLK에 연동되었기 때문에 각 모듈들을 쉽게 연동할 수 있으며 ARMulator의 프로파일링 기능을 이용하여 시스템의 성능을 쉽고 정확하게 예측할 수 있다. AHB와 APB 버스는 Bridge 블록을 통하여 서로 연결되며 각 버스는 실제 전송 프로토콜보다는 기능과 전송 시간이 정확하도록 구현되었다. AHB 버스에 3개의 마스터가 연결되어 있기 때문에 우선순위-기반 버스 조정자가 구현되었다.

구현된 하드웨어 모델들의 동작은 삼성의 응용 프로세서인 S3C2440의 하드웨어 IP과 유사하며 제어를 위한 소프트웨어 인터페이스를 동일하게 하였다. 대부분의 삼성 칩은 S3C2440과 유사한 하드웨어 IP들을 가지고 있기 때문에 구축된 가상 실습 키트는 삼성에서 제작된 ARM 코어 기반 칩을 사용하는 시스템의 교육에 더욱 효과적으로 사용될 수 있다.

3. 사용자 인터페이스 및 실시간 운영체제 이식

S3C2440을 참조하여 확장된 ARMulator 환경을 임베디드 시스템 보드의 형태로 구축하고 학습자가 실제 하드웨어의 동작을 확인하기 위해서는 확장 하드웨어 IP들의 기능과 동작을 확인할 수 있는 주변장치들이 필요하다. 이를 위하여 LCD 패널, LED 표시장치, UART 연동 인터페이스, 타이머 동작 표시기를 윈도우즈 응용 프로그램으로 구현하였다. 확장 환경의 LCD 콘트롤러는 윈도우즈 IPC (Inter-Process Communication)를 이용하여 LCD 패널과 연결하였으며 LCD 패널은 LCD 콘트롤러가 보내주는 이미지 데이터를 화면에 표시한다. LCD 패널에 이미지가 보이게 하기 위해서는 LCD 콘트롤러의 제어 레지스터를 설정하여야 한다. 설정하는 내용은 LCD 패널 형태, 색 표현 형식, 메모리 내의 이미지 버퍼 위치 등이다. LCD 콘트롤러가 동작하면 프로세서는 이미지 버퍼에 픽셀 데이터를 채우고 LCD 콘트롤러의 내부 DMAC가 주기적으로 픽셀 데이터를 LCD 패널에 전달하여 LCD 패널을 구동하는 형태로 동작한다. 확장 환경의 GPIO 포트의 일부분을 가상적으로 4개의 LED 표시 프로그램과 연결을 시켜서 GPIO를 통한 on-off 제어의 내용이 LED에 표시되도록 하였다. UART 모듈 동작의 확인은 실제 컴퓨터의 COM 포트에 UART IP 모듈이 직접 데이터를 송수신할 수 있도록 장치 제어 윈도우즈 API를 이용하여 COM 포트를 제어하도록 하였다. 따라서 UART 모듈의 동작은 실제 직렬 포트에 연결된 다른 시스템의 터미널 프로그램을 통해서 확인할 수 있다. 확장된 타이머 IP는 PWM (Pulse Width Modulation) 신호를 출력할 수 있다. 타이머가 동작하면서 발생시키는 PWM 신호는 로그 파일 형태로 기록되며 GTKWave라는 프로그램을 이용하여 파형을 확인할 수 있다. [그림 3]은 최종적으로 구축된 가상 실습 키트 환경을 도시하고 있다.

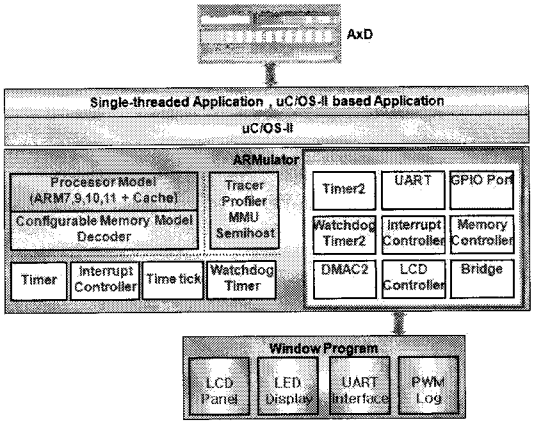


그림 3. 개발된 가상 실습 키트 환경

개발된 가상 실습 키트는 ARM 사의 ADS 1.2에 포함되어 있는 ARMulator 환경을 사용하기 때문에

ADS1.2의 AxD 디버거를 통하여 연결할 수 있다. ADS1.2의 도구들을 이용하여 만든 수행 이미지는 AxD를 통해 메모리 모델에 적재하고 AxD의 디버깅 기능을 이용하여 수행과 디버깅을 진행한다. 프로그램의 실행 결과는 제공되는 주변장치 프로그램을 이용하여 확인할 수 있다. 가상 실습 키트를 이용하여 실시간 운영체제에 대한 학습이 가능하도록 실시간 운영체제인 MicroC/OS-II[16]를 가상 실습 키트에 이식하였다. MicroC/OS-II는 마이크로프로세서 및 마이크로콘트롤러를 위한 작은 크기의 이식성과 적용성이 뛰어난 선점형 실시간 멀티태스크 커널로서 임베디드 시스템 운영체제 교육에 많이 사용되고 있다.

4. 구현된 가상 실습 키트의 검증

4.1 버스 동작의 검증

가상 실습 키트가 새로운 하드웨어 IP의 추가로 쉽게 확장될 수 있으며 정확한 성능을 측정하기 위해서 CPU 클록과 연동되는 AHB 버스를 구현했으며 이 버스의 기능 확인을 위해 임의의 Load/Store 명령을 실행하는 간단한 ARM 실행 프로그램을 실행하였다. 실행 중 내부 동작을 확인할 수 있도록 ARMulator의 호스트 인터페이스를 통해 AxD의 RDI log 창에 매 버스 사이클 마다 버스가 실행한 전송에 대한 정보가 표시되도록 하였다. [그림 4]는 버스 전송 로그를 캡처한 것이다.

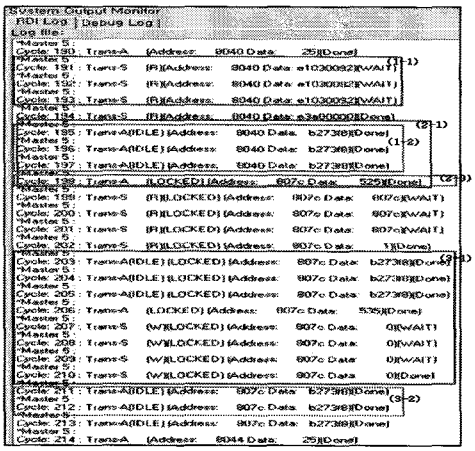


그림 4. 검증을 위한 버스 전송 로그

버스의 내부 상태에 따라 다양한 전송을 보여주고 있으며 버스의 동작이 정확함을 검증하였다. [그림 4]에서 'Cycle 191 ~ 193' (1-1)은 Wait response 전송이고 'Cycle 195 ~ 197' (2-1)은 순수 IDLE cycle 전송이다. 또 'Cycle 203 ~ 210' (3-1)은 Swap 명령에 따른 LOCK 전송이다. 이외에 DMA와 프로세서가 동시에 동작하는 실행을 통하여 DMA와 버스 조정자의 동작을 확인하였다.

4.2 하드웨어 IP와 사용자 인터페이스 검증

구현된 가상 실습 키트의 하드웨어 IP들인 인터럽트 콘트롤러, 타이머, LCD 콘트롤러 및 LCD 패널, GPIO 포트 및 LED 표시, UART의 동작을 확인하기 위하여 이들 IP 및 주변장치를 모두 사용하는 디지털시계 프로그램을 작성확인하기행 시켰다. 프로그램은 실행 중에 UART를 통해 연결된 터미널 프로그램에 메시지를 보내고 타이머동작을 확인하일정 간격으로 인터럽트동작발생시켜 인터럽트 핸들동작을호출한다. 핸들동는 GPIO에 연결된 LED동작순서대로 On연결갰갰시키며 LCD 패널에 표시 O에간을 구하고 에간의 숫자순서맞는 이미지를 이미지 버퍼에 저장한다. LCD 콘트롤러는 이 이미지들을 LCD 패널에 전달확인하에간이 표시되게 한다. [그림 5]는 테스트 프로그램 실행 중 쌓결M 포트를 통한 터미널 프로그램의 출력과 LCD/LED 화면 인터페이스를 캡처한 그림이다. 본 프로그램의 실행을 통해 각 장치의 동작이 정상적으로 이루어짐을 확인 할 수 있었다.

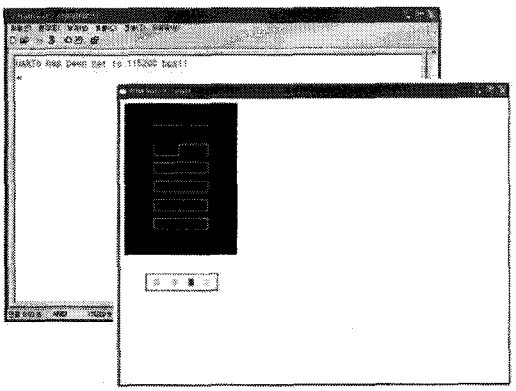


그림 5. UART와 LCD/LED 표시 인터페이스

IV. 가상 실습 키트의 교육적 활용

구축된 임베디드 시스템 교육용 가상 실습 키트를 사용하여 다음과 같은 실습 내용의 구성이 가능하다.

- ARM 시스템 개발 환경: 가상 실습 키트는 ADS1.2를 기반으로 구축되었기 때문에 실습을 진행하기 위해서는 ARM 시스템 소프트웨어 개발 환경에 익숙해 져야 하며 ARM 시스템 소프트웨어 개발을 위한 어셈블러, 컴파일러, 링커 등의 사용과 동작, 이미지 제작과 프로그램 실행을 위한 프로그램 레이아웃 구성에 대한 실습을 진행할 수 있다. 또한 프로그램의 디버깅을 위한 AxD 디버거의 사용 방법 및 디버거의 세부 기능을 학습할 수 있다.

- 어셈블리언어, C/C++ 언어를 이용한 하드웨어 IP 제어: 구현된 가상 실습 키트는 삼성의 S3C2440 하드웨어 IP 중에서 마이크로프로세서 시스템 학습에 필수적인 IP들을 구현하고 있다. ARM 어셈블리언어 또는 C/C++ 언어를 이용한 하드웨어 IP 제어를 통해 하드웨어 IP들의 동작과 제어 방법을 실습할 수 있고 부수적으로 ARM 어셈블리언어와 고급언어를 사용한 하드웨어 제어 방법을 학습할 수 있다.

- 마이크로프로세서 시스템 구성 및 성능: ARMulator는 프로세서의 종류와 속도, 메모리의 구성 및 접근 시간을 설정할 수 있다. 구현된 하드웨어 IP들은 CPU 클록에 연동되어 동작하기 때문에 ARMulator에서 제공하는 프로파일링 기능을 이용하여 프로세서, 메모리, IP들의 시스템 구성에 따른 성능에 대한 다양한 탐색이 가능하다.

- 실시간 운영체제: 가상 실습 키트에 MicroC/OS-II를 이식하였기 때문에 운영체제 API를 이용한 다중-쓰레드 프로그램과 디바이스 드라이버를 작성할 수 있다. MicroC/OS-II에서 제공하는 메시지, 큐, 메일박스, 세마포어와 같은 기능들에 대한 실습이 가능하다.

위와 같은 주제 외에도 확장 가능한 ARMulator 환경을 이용하여 학습자에게 특정 하드웨어 IP를 모델링하여 키트에 추가하는 실습을 통해 IP에 대한 정확한 동작과 시스템 구성에 대한 이해를 높일 수 있다. 또한 필요시에 하드웨어 IP를 추가하여 특정 과목에 적합한 가상 실습 키트를 구축할 수도 있으며 개발된 가상 실습 키트는 실제 하드웨어 동작을 거의 똑같이 구현하고 있어서 시스템 초기화 코드의 작성과 같은 실습도 가능하다.

V. 결 론

본 논문에서는 다양한 임베디드 시스템 실습 교육을 통합된 환경에서 수행할 수 있는 가상 개발 환경의 사용을 제안하였다. 또한 기존 상용 가상 개발 환경을 대체할 수 있는 현실적인 가상 실습 키트의 설계 및 구현에 대하여 기술하였다. 구현된 가상 실습 키트는 기존 하드웨어 실습 키트의 단점인 설치 및 유지보수의 비용을 줄일 수 있으며 다양한 임베디드 시스템 교육을 통합된 환경 하에서 진행할 수 있는 장점이 있다. 하드웨어 및 소프트웨어 동작을 시뮬레이션 환경에서 자세히 관찰할 수 있기 때문에 교육 효과 측면에서 우수하며 안전한 실습 환경을 구축할 수 있다.

구현된 키트는 ARM사의 소프트웨어 개발 환경인 ADS 1.2에 포함되어 있는 ARMulator를 이용하였으며 ARM920T 프로세서 코어의 시뮬레이션 환경에 삼성의 휴대형 단말장치를 위한 S3C2440의 주요 하드웨어 IP들을 모델링하여 추가하였다. 또한 임베디드 시스템 보드에 필요한 주변 장치 및 사용자 인터페이스를 윈도우즈 프로그램 형태로 추가하였다. 하드웨어 IP들은 CPU 클록에 동기 되는 AHB 버스 및 APB 버스에 연결이 되어서 사이클 단계의 정밀한 시뮬레이션을 이용한 실습이 가능하다. 기존의 컴퓨터구조 관련 시뮬레이션 도구들이 추상화 되거나 특정 주제에 특화된 실습만 가능하였던 것에 반하여 구축된 가상 실습 키트를 이용하여 시스템 개발 환경, 하드웨어 IP 제어, 마이크로프로세서 구성 및 성능에 관련된 주제에 대한 다양한 실습을 진행할 수 있다. 가상 실습 키트에 실시간 운영체제인 MicroC/OS-II를 이식하여 실시간 운영체제에 대한 실습도 가능하다. 전체 시스템의 정확한 동작은 주요 하드웨어 IP를 모두 사용하는 디지털시계 프로그램의 수행으로 확인하였다. 새로운 하드웨어 IP가 필요할 경우

에는 ARMulator 환경에서 제공하는 모듈 제작 환경을 이용하여 모델링하고 구현된 AHB와 APB 버스에 쉽게 연결할 수 있어서 가상 실습 키트를 새로운 실습 환경에 맞추어 지속적으로 확장할 수 있다.

비록 구현된 가상 실습 키트가 임베디드 시스템 교육을 위한 대부분의 실습을 수행할 수 있지만 사용자 인터페이스는 개선의 여지가 많다. 실제 하드웨어 키트와 유사한 모습으로 사용자 인터페이스가 제작되어야 하며 하드웨어 실습 환경을 설정할 시에 필요한 전원 및 선들의 연결과 같은 동작도 애니메이션과 같은 형태로 구현하여 보다 실감적인 가상 실습 키트가 될 수 있도록 해야 할 것이다.

참 고 문 헌

[1] S. Hussmann and D. Jensen, "Crazy car race contest: Multicourse design curricula in embedded system design," IEEE Trans. Educ., Vol.50, No.1, pp.61-67, 2007(2).

[2] R. E. Seviora, "A curriculum for embedded system engineering," ACM Trans. Embed. Comput. Syst., Vol.4, No.3, pp.569-586, 2005(8).

[3] J. O. Hamblen, "Using a low cost SoC computer and a commercial RTOS in an embedded systems design course," IEEE Trans. Educ., Vol.51, No.3, pp.356-363, 2008(8).

[4] S. Nooshabadi and J. Garshide, "Modernization of teaching in embedded systems design-An international collaborative project," IEEE Trans. Educ., Vol.49, No.2, pp.254-262, 2006(5).

[5] V. Reddi, A. Settle, D. Connors, and R. Cohn, "PIN: A binary instrumentation tool for computer architecture research and education," Proc. of the 2004 workshop on Comp. Arch. Educ., 2004(6).

[6] L. Null and J. Lobur, "MarieSim: The MARIE computer simulator," ACM Jour. of Educ. Reso. in Comp., Vol.3, No.3, pp.1-29, 2003(6).

[7] G. Wolfe, W. Yurcik, H. Osborne, and M. Holliday, "Teaching computer organization/architecture with limited resources using simulators," ACM SIGCSE Bulletin Vol.34, No.1, pp.176-180, 2002.

[8] B. Nikolic, Z. Radivojevic, J. Djordjevic, and V. Milutinovic, "A survey and evaluation of simulators suitable for teaching courses in computer architecture and organization," IEEE Trans. Educ., Vol.52, No.4, pp.449-458, 2009(11).

[9] A. Stojkovic, J. Djordjevic, and B. Nikolic, "WASP: A web based simulator for an educational pipelined processor," Int. J. Elect. Eng. Educ., Vol.44, No.3, pp.197-215, 2007(7).

[10] L. Ivanov and J. Mallozzi, "A hardware/software simulator to unify courses in the computer science curriculum," Jour. of Computing Sciences in Colleges, Vol.19, No.5, pp.238-248, 2004(5).

[11] Synopsis Corp. Virtual Platform: http://www.synopsys.com/Tools/SLD/Pages/default.aspx.

[12]http://www.carbondesignsystems.com/Products/SoCDesigner.aspx.

[13]http://www.mentor.com/products/esl/design_verification/vista_architect/.

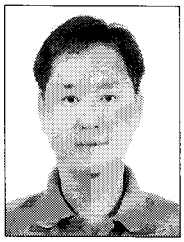
[14]http://www.arm.com/products/DevTools/RealViewSoftwareDevelopment.html.

[15] Samsung Electronics, "S3C2440A Users Manual Revision 1.0," Mar. 2004.

[16] Jean J. Labrosse. *MicroC/OS-II Real Time Kernel 2/E*, R&D Technical Books, 2002.

저 자 소 개

조 상 영(Sang-Young Cho) 종신회원



- 1988년 2월 : 서울대학교 제어계측공학과(공학사)
- 1990년 2월 : KAIST 전기전자공학과(공학석사)
- 1994년 2월 : KAIST 전기전자공학과(공학박사)
- 1994년 3월 ~ 1995년 1월 : KAIST 정보전자연구소 위촉연구원
- 1995년 1월 ~ 1996년 2월 : 미국 UC Irvine ECE Post Doc.
- 1996년 3월 ~ 1997년 2월 : 삼성전자 선임연구원
- 1997년 3월 ~ 현재 : 한국외국어대학교 컴퓨터공학과 교수

<관심분야> : 임베디드 시스템, 가상 개발 환경